AMENDMENTS TO THE CLAIMS

1. (Previously Presented) A method for automatically generating a description of a data exchange format based on computer program source code expressed in a source language, the method comprising the computer-implemented steps of:

receiving, from a source code file, comment data including first data indicating a parameter of the data exchange format, wherein the comment data is ignored by a source code processor of the source language;

receiving from the source code file second data, associated with the comment data, indicating a statement that defines a class of data objects in the source language;

wherein the data exchange format defines, using symbolic tags, a hierarchical structure of data that is exported from, and imported to, data objects of the class of data objects;

automatically generating, based on the first data and the second data, third data that describes the data exchange format, wherein the third data comprises instructions defining a mapping between attributes of the class of data objects and elements of the data exchange format; and

generating a marshalling module to convert a data object of the class of data objects into a data item of the data exchange format as described by the third data.

2. (Previously Presented) A method as recited in Claim 1, wherein generating the marshalling module is based on the first data and the second data.

3. (Previously Presented) A method as recited in Claim 1, wherein generating the marshalling module is based on the third data.

4. (Previously Presented) A method as recited in Claim 1, further comprising generating, based on the first data and the second data, a de-marshalling module to derive the data object of the class of data objects from the data item of the data exchange format as described by the third data.

5. (Previously Presented) A method as recited in Claim 1, further comprising generating, based on the third data, a de-marshalling module to derive the data object of the class of data objects from the data item of the data exchange format as described by the third data.

6. (Original) A method as recited in Claim 1, wherein the third data is formatted according to a database query language.

7. (Original) A method as recited in Claim 1, wherein the third data is formatted according to a symbolic markup language.

8. (Original) A method as recited in Claim 1, wherein the third data is formatted according to extensible markup language (XML).

9. (Original) A method as recited in Claim 1, wherein the third data comprises one or more statements in an XML schema document.

10. (Original) A method as recited in Claim 1, wherein the third data is one or more statements in an XML document type definition (DTD) document.

11. (Original) A method as recited in Claim 1, wherein the third data is one or more statements in an XML document type definition (DTD) document, and wherein the parameter is at least one of a root element associated with an entire DTD document, an element and an attribute of an element.

12. (Original) A method as recited in Claim 1, wherein the third data is one or more statements in an XML document type definition (DTD) document, and wherein the first data includes one or more properties of the parameter.

13. (Previously Presented) A method as recited in Claim 1, wherein the source language is JAVA.

14. (Previously Presented) A method as recited in Claim 1, wherein the source language is JAVA, and wherein the first data includes a tag for an automated JAVA documentation system.

15. (Previously Presented) A method as recited in Claim 1, wherein the source language is JAVA, wherein the first data includes a tag for an automated JAVA documentation system, and wherein the tag is a user-defined tag for the JAVA documentation system.

16. (Previously Presented) A method as recited in Claim 1, wherein the source language is JAVA, wherein the first data includes a tag for an automated JAVA documentation system, wherein the tag is a user-defined tag for the JAVA documentation system, and wherein said step of generating the third data is performed by a user-defined routine invoked by the automated JAVA documentation system in response to the tag.

17. (Canceled)

18. (Canceled)

19. (Canceled)

20. (Canceled)

21. (Previously Presented) A computer-readable medium storing one or more sequences of instructions for binding a data exchange format with an application having source code in a particular language, which instructions, when executed by one or more processors, cause the one or more processors to carry out the steps of:

receiving, from a particular file that includes the source code, comment data including first data indicating a parameter of the data exchange format, wherein the comment data is ignored by a source code processor of the particular language;

receiving from the particular file second data, associated with the comment data, indicating a statement that defines a class of data objects in the particular language;

wherein the data exchange format defines, using symbolic tags, a hierarchical structure of data that is exported from, and imported to, data objects of the class of data objects;

generating, based on the first data and second data, third data for configuring the data exchange format, wherein the third data comprises instructions defining a mapping between attributes of the class of data objects and elements of the data exchange format; and

generating a marshalling module to convert a data object of the class of data objects into a data item of the data exchange format as described by the third data.

22. (Currently Amended) An apparatus for binding a data exchange format with an application having source code in a particular language, comprising:

means for receiving, from a particular file that includes the source code, comment data including first data indicating a parameter of the data exchange format, wherein the comment data is ignored by a source code processor of the particular language;

means for receiving from the particular file second data, associated with the comment data, indicating a statement that defines a class of data objects in the particular language;

wherein the data exchange format defines, using symbolic tags, a hierarchical structure of data that is exported from, and imported to, data objects of the class of data objects;

means for generating, based on the first data and second data, third data for configuring the data exchange format, wherein the third data comprises

instructions defining a mapping between attributes of the class of data objects and elements of the data exchange format; and

means for generating a marshalling module to convert a data object of the class of data objects into a data item of the data exchange format as described by the third data.

23. (Previously Presented) An apparatus for binding a data exchange format with an application having source code in a particular language, comprising:

a processor;

one or more stored sequences of instructions which, when executed by the processor, cause the processor to carry out the steps of:

receiving, from a particular file that includes the source code, comment data including first data indicating a parameter of a data exchange format, wherein the comment data is ignored by a source code processor of the particular language;

receiving from the particular file second data, associated with the comment data, indicating a statement that defines a class of data objects in the particular language;

wherein the data exchange format defines, using symbolic tags, a hierarchical structure of data that is exported from, and imported to, data objects of the class of data objects;

generating, based on the first data and second data, third data for configuring the data exchange format, wherein the third data comprises instructions defining a mapping between attributes of the class of data objects and elements of the data exchange format; and

generating a marshalling module to convert a data object of the class of data objects into a data item of the data exchange format as described by the third data.

24. (Previously Presented) A method for automatically generating a description of a data exchange format based on a JAVA source code of an application, the method comprising the computer-implemented steps of:

receiving, from a JAVA source code file, comment data that comprises one or more tags of the data exchange format and one or more parameters that are associated with the one or more tags, wherein the comment data is ignored by a source code processor of the JAVA language;

receiving, from the JAVA source code file, one or more statements that define a class of data objects in the JAVA language, wherein the class of data objects is associated with the comment data;

wherein the data exchange format expresses, in eXtensible Markup Language (XML), the structure of data that is exported from, and imported to, data objects of the class of data objects;

based on the one or more tags, the one or more parameters, and the one or more statements, a JAVADOC documentation system generating an XML Document Type Definition (DTD) document that describes the data exchange format, wherein the XML DTD document comprises instructions defining a mapping between attributes of the class of data objects and elements of the data exchange format;

based on the one or more tags, the one or more parameters, and the one or more statements, generating a marshalling module for converting a data object of the class of data objects into a data item of the data exchange format as described by the XML DTD document; and

building an executable version of the application based at least on the JAVA source code and the marshalling module.

25. (Previously Presented) A method as recited in Claim 24, wherein:

the one or more tags are user-defined JAVADOC tags; and

each of the one or more parameters is any one of a root XML element associated with the entire XML DTD document, an XML element of the data exchange formal, and an attribute of an XML element of the data exchange format.

26. (Canceled)

27. (Previously Presented) A method as recited in Claim 24, further comprising:

based on the one or more tags, the one or more parameters, and the one or more statements, generating a de-marshalling module for deriving the data object of the class of data objects from the data item of the data exchange format as described by the XML DTD document;

wherein building the executable version of the application is further based on the de-marshalling module.